What Happens to Your Data After Its Shot

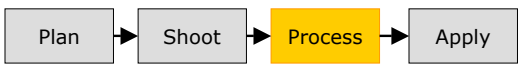
Michael Gleicher
University of Wisconsin- Madison

Taylor Wilson
House of Moves Studios

Topics

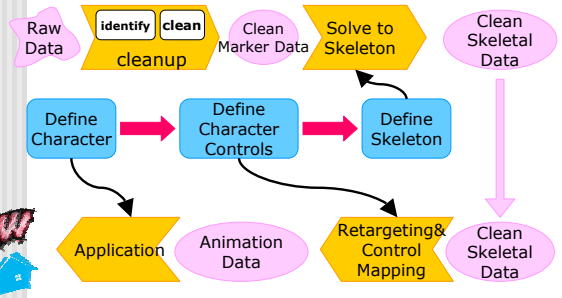
- Issues we must deal with
- The methods we might throw at it
- Some things to watch out for

Mocap Pipeline



- This talk is about Processing
  - A pipeline unto itself
- The process as a response to the issues
- Pushing problems as early as possible

Processing Pipeline



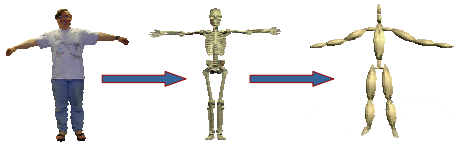
What Comes In? What Goes Out?

- Raw 3D data comes in
  - Marker positions
  - Marker orientations (sometimes)
- Ultimate Output: Animation
  - Whatever you need to drive the character
- Intermediate Step:
  - Abstract representation of what motion really happened

Begin with the End in Mind

- The real goal is to get the motion that you want
- If what you need isn't in your original data, forget it!
  - What motions?
  - What degrees of freedom?
  - What quality level?

Abstraction of Human Motion



- Question of Approximating DOF's
  - (morning talk)
- Some number of connected, rigid pieces
  - (usually)

Representations of Motion

- Angle vs. positional data
- Global vs. relative
- Hierarchical vs. non-hierarchical
- Skeletal vs. Non-Skeletal

Good Points of Hierarchical Skeletons

- Enforce key constraints
  - Connected segments
  - Rigid limbs
- Fewer Dof's
  - Only store angles between segments
- Easy for skinning
  - Local coordinate systems defined

Bad Points of Hierarchical

- Need 3D rotations
- Coupled parameters
- End effector controls require IK
- Forces rigidity
- Problems with reference
  - Different ways of defining things

How to Maximize Good / Minimize Bad

- Custom character setup (have right DOFs)
- Well chosen joint sets (placement and type) and controls (IK / FK)
- Good:
  - make characters that animator can control
- Bad: no uniformity/standardization
  - important if motion from outside source
  - important if want to build libraries / reuse motions
  - Everybody has a different skeleton

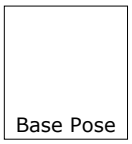
How do skeletons differ?

- Obvious ways?
  - Topology
    - number of bones
    - Connectivity of bones
  - Joint Types
  - Bone lengths
  - Anatomical / skin relations
    - Is spine in middle of body, or up the back?

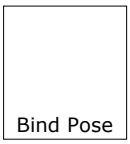
Subtle Skeletal Differences

- What to measure angles with respect to
  - Doesn't matter, as long as we agree
- Poses (design of a skeleton)
  - Zero Pose / Base Pose
  - Dress or Binding pose
  - Frankenstein Pose
  - Da Vinci Pose
  - Rest Pose (real pose of actor)
- Need to figure out how to get between these

Target Poses



AKA Zero Pose. What happens when all joints are set to zero



AKA Dress Pose. What happens when all joints are set to zero

These poses do not say what the pose looks like!

Reference Poses

All limbs as vertical as possible

Arms Horizontal, Legs Spread

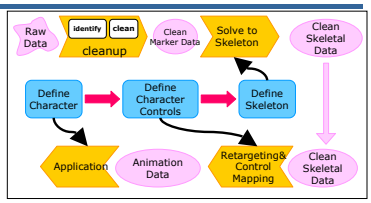
How the actor stands at rest

These pose names say what the pose looks like, not how to get them!

From Here to There

- We know what we want (skeletal data)
- Might want something else!
  - Constraint data (drive IK chains)
  - Clean marker data
  - Whatever makes our character go!
- We know what we have ("raw" data)

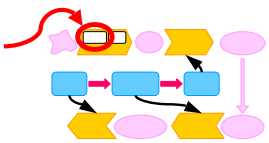
Issues



- Labeling / correspondence
- Noise (and gap filling)
- Reconstruction / conversion to skeletal
- Application to target character

Correspondence / Labeling

First part of cleanup

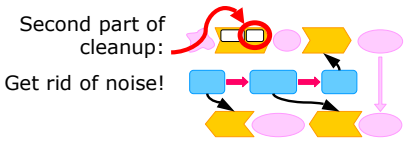


- Which marker is which body part?
- Sometime solved by hardware for you
  - Active markers solve correspondence
  - Only really a problem for passive optical

Software Solutions

- Coherence gets most cases
  - Works except when you get massive occlusions/many characters
- Marker "groupings"
- Assumed rigid relationships
- Marker group/joint limits
- Skeletal feedback

Noise



- Definition: Unwanted randomness
- Goal: get rid of it
- Problem: don't know what "it" is

Where's the Noise?

- Sometimes identification is easy:
  - Clearly wrong (foot through floor)
  - Marked wrong (missing data - gaps)
- More often, need to guess
- Magnetic has "gaps" - just harder to identify

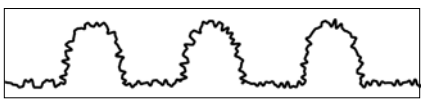
Noise Detection

- Use heuristics and rules of thumb to identify noise
- Use info about which body part as a discriminator
  - Extremeties are more likely to have sharp movement
- "Speed" of the movement affects how prevalent noise is
  - Visual signal/noise ratio decreases as movement gets slower

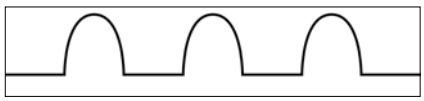
Mocap Noise Misconception

- Things in the world don't change that fast (have high freq)
- If there are high freqs, must be noise
- Get rid of high freqs (quick changes)
- Low-Pass Filter (LPF) easy (weighted average, FIR, ...)

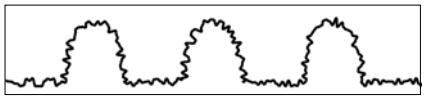
Low-Pass Filters vs. Noise



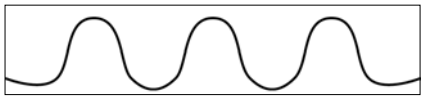
- We want to remove the noise, to get back a signal that looks like



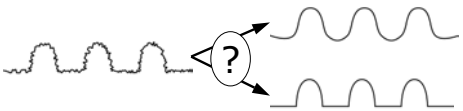
Low-Pass Filters vs. Noise



- Getting Rid of High Frequencies does not just eliminate noise
- Leaves a "soggy" look

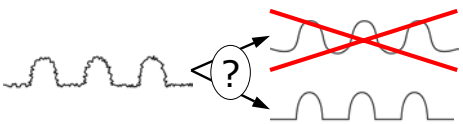


High Frequencies



- PROBLEM: High frequencies can be important!
  - Getting rid of them makes motion look soggy
- ANSWER: Do not over-apply LPF
  - How much is enough?
  - Use a little LPF

Treating Mocap Noise



- Small amounts of Low-Pass Filtering
- Noise modeling
- Adaptive filters
- Non-linear filters
- Hybrid solutions

Now we know where markers are

- Convert this to skeletal
- Basic Issues:
  - Inexact fit
  - Humans are not rigid skeletons
  - Markers imperfect - have momentum, noise
  - Markers on "skin", not directly connected to skeleton

Complexities of Skeletal Representation

- Can't just measure
  - (even x-rays wouldn't help, no real "joints"

Picture of Human shoulder vs. CG shoulder

- Abstraction
- Don't know parameters
- Need to know skeleton and relation of skeleton to markers

Solving Problems Early

- Choose good marker sets
  - Redundant, minimal sliding, close to bone
- Get mechanics as good as possible (glue markers on well!)
- Get low-level data as good as possible
  - Although, high level can be used to solve low-level problems
- Redundancy

Strategies for Skeletal Conversion

- Optimization
- Direct geometric

Optimization

- Get skeleton as close as possible to markers
- Need some metrics for how attached

Optimization Pros

- Distribute error over whole skeleton
- Can solve for parameters simultaneously
- Can set up complex relationships between markers and bones
- Possibly use spacetime to insure coherence

Optimization Cons

- Complex
- Global (one mistake can mess up everything)
- Harder to implement / understand / diagnose
- Possibly too many parameters
- Need starting points
- Can only use differentiable (non-robust) metrics

Geometric Solutions

- Worth learning about (intuitions, easy, ...)
- Use simple building blocks
- Why/why not?

Geometric Solution Pros

- Local
- Simple
- Use robust metrics
- Separate parameter finding and value finding

Geometric Solution Cons

- Local
- Separate parameter finding and value finding
- Bad data can cause complete failures
- Requires cleverness in setting up processes

A Simple Marker Processor

- A walk through the geometric process
- Why?
  - Still a useful way to do things
  - Give insights
  - Show problems that must be addressed no matter what
  - Much easier to implement than other approaches

Basics

- Markers as oriented frames
  - Allows for magnetic (6dof) data too
- Operations change these frames
- Strictly procedural
  - A sequence of steps that is always followed

Basic Operations

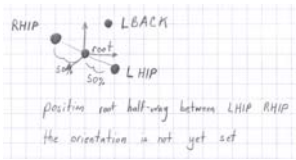
- Absolute w/two other markers
  - new
  - position (absolute)
  - move (relative)
  - orient (absolute)
  - rotate (relative)
  - copy

Process Walkthrough - Multiple Phases

- Add markers to approximate joint centers
- Estimate bone lengths (optional)
- Enforce bone lengths
- Factor to joint angles

Step 1: Add a root node

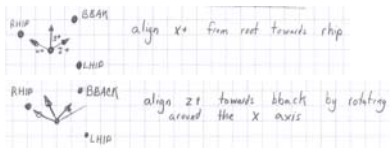
```
position root RHIP LHIP .5
```



- Place a new marker (root) halfway between RHIP and LHIP

Step 2: Orient root node

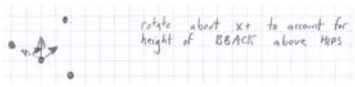
Orient root RHIP x+ LBAK z+



- Align the coordinate system such that the X+ axis points towards RHIP, and the Z+ axis points toward LBAK

Step 3: Orient Root Node

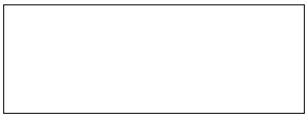
rotate root x+ 45



- Rotate coordinate system around X axis by a specified amount

Step 4: place a hip joint center

Position rhips root RHIP 0
Move rhips 6 0 0



- Place the new joint center at the root position, then move it along the local axis by a specified amount

Step 5 place a knee joint

```
Position rknee RKNE RANK 0
Orient rknee RANK y- RHIP z+
Move rknee -2 0 0
```

- Position the knee at the knee marker
- Orient the axes by neigboring makers
- Use an absolute position offset

In Practice…

- Direct / Geometric methods still do get used
- Require cleverness to define series of steps
- Richer operation sets lead to more convenience, and quality

How do you apply this data?

- See the other talks…